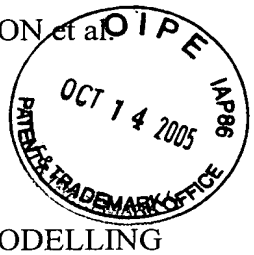
IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: ANDERSON et al.

Application No.: 10/027,835

Filed: 12/20/2001

Title: SIMULATION AND MODELLING METHOD AND APPARATUS



Attorney Docket No.: MADDP001

Examiner: PROCTOR, Jason Scott

Group: 2123

Confirmation No.: 2571

CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the U.S. Postal Service with sufficient postage as first-class mail on October 11, 2005 in an envelope addressed to the Commissioner for Patents, P.O. Box 1450 Alexandria, VA 22313-1450.

Signed: _____
Laura M. Dean

TRANSMITTAL OF CERTIFIED PRIORITY DOCUMENTS

Mail Stop Amendment
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

Transmitted herewith is the certified copy of the priority document for the above-referenced patent application, SIMULATION AND MODELLING METHOD AND APPARATUS, Australian Patent Application No.PQ 7106.

The Commissioner is authorized to charge any fees that may be due to Deposit Account No. 50-0388 (Order No. MADDP001).

Respectfully submitted,

BEYER WEAVER & THOMAS, LLP

Alan S. Hodes
Registration No. 38,185

P.O. Box 70250
Oakland, CA 94612-0250

Australian Government

Patent Office
Canberra

I, LEANNE MYNOTT, MANAGER EXAMINATION SUPPORT AND SALES hereby certify that annexed is a true copy of the Provisional specification in connection with Application No. PQ 7106 for a patent by THE COMMONWEALTH OF AUSTRALIA as filed on 20 April 2000.

WITNESS my hand this
Thirteenth day of September 2005

LEANNE MYNOTT
MANAGER EXAMINATION SUPPORT
AND SALES

COPY OF
PRIORITY DOCUMENT

THE COMMONWEALTH OF AUSTRALIA

ORIGINAL

AUSTRALIA
PATENTS ACT 1990

PROVISIONAL SPECIFICATION FOR THE INVENTION ENTITLED:

"SIMULATION AND MODELLING METHOD AND APPARATUS"

This invention is described in the following statement:

This invention relates generally to simulation modelling and an embodiment is provided relating to the simulation of airborne missiles comprising for example Air-to-Air (AAM) and Surface-to-Air (SAM) Missiles.

BACKGROUND

The science of modelling and simulation, by definition, requires compromise. Those skilled in the art use the term "fidelity" to describe how closely the model mimics reality. If a model provides output which is far from reality, the fidelity is low, and when a model provides output which is close to reality, the fidelity is considered to be high.

Of necessity, models rely on the formulation of mathematical equations having one or more variables, wherein the rule of thumb is that the larger the number of variables and the more complex the equation, the more likelihood there is of good fidelity.

Sometimes simple models are better than none, as the model can then be said to be generic to a particular class of conditions, and sometimes very simple models are forced upon designers, because calculation time or power constraints apply.

However once a more complicated model is developed, it becomes less generic and the number of circumstances to which it can apply reduces to the extent that separate models are required for each circumstance if the highest fidelity is to be achieved.

Such an approach has the clear advantage that the fidelity of each simulation is good, but the disadvantage is that as fidelity increases so does the computational requirements as well as the quantity and accuracy of data used to support the model.

However, when large amounts of data are required for each model and circumstance, it can very quickly become unwieldy to create and maintain the data in a database. This complexity and difficulty increases more so for distributed application of the

model particularly when the users of the application are widely separated geographically.

One way of better managing the association of data with particular circumstances, is to use a mechanism in the generic model which call up data specific to the circumstance for execution by the one or more equations available to the model.

Thus modelling of this type has its advantages but clearly maintains the disadvantage of creating a specific rigidly structured key word database for each required circumstance.

The invention described and disclosed in the specification aims to reduce or obviate the problems described above.

The invention is directed to a simulation approach which comprises two elements, the first element comprises a generic mathematical model used to simulate a particular environment and a particular class of objects interacting with that environment, and the second element comprising a set of user defined data files which contain not only predetermined parameters for use by the first element but which also contains executable mathematical equations and respective data relating to a specific object entering the environment which add to and complement the generic mathematical model so as to produce, when operating together, a simulation of a particular environment's interaction with a particular object.

BRIEF DESCRIPTION OF THE INVENTION

In a broad aspect of the invention a simulation arrangement comprises an interpretation and display module having a generic calculation means to calculate an object movement and an interface means for loading and interpreting object data;

a data storage means containing data files which contain numeric data, syntactic key words and user defined algorithmic expressions;

a virtual machine which includes means to load data from the data files for execution so as to assign a value to a parameter which is then provided to said interpretation and display module for use in the calculation of the object movement for a given simulation time-step.

Specific embodiments of the invention will now be described in some detail with reference to and illustrated in the accompanying figures. These embodiments are illustrative, and not meant to be restricted to the scope of the invention. Suggestions and descriptions of other embodiments may be included but they may not be illustrated in the accompanying figures or alternatively features of the invention are shown on the figures, but not described in the specification.

BRIEF DESCRIPTION OF THE FIGURES

Fig. 1 depicts the functional block diagram of a missile engagement and coverage analysis arrangement;

Fig. 2 depicts the functional block diagram of the functions of the MECA engagement class;

Fig. 3 provides a pictorial representation of the MECA data bus;

Fig. 4 provides a representation of the data exchange between the missile model, the data bus and the data manager;

Fig. 5 depicts a pictorial representation of the missile model and its interaction with portions of the engagement model;

Fig. 6 depicts a portion of an example MECA missile data file; and

Fig. 7 depicts two dimensional representation of a one-on-one simulation.

Table 1 depicts the program start-up sequence; and

Table 2 depicts selected code of a program start-up and run simulation associated with a one-on-one simulation.

DESCRIPTION OF AN EMBODIMENT OF THE INVENTION

For the purposes of pre-purchase evaluation and operational command training it is useful to simulate, in the digital environment, the performance of AAM's and SAM's. Simulation, as long as it has adequate fidelity, is particularly useful in the design process, and helps to reduce the time to evaluate new missiles in various scenarios, including active comparison with existing missiles. Simulation can also assist aircraft pilot and gunnery (air and ground based) training without the expense of real flying and missile launches which have unacceptable risks.

Missile engagement and coverage analysis is the common term used to describe the types of simulation described as an embodiment in this specification. However, any type of modelling and simulation will benefit from the use of a data file which can contain numeric data, syntactic key words or user defined algorithmic expressions which is termed herein as the "SMART DATA" approach of the invention.

The performance of AAM's and SAM's vary greatly dependent upon their launch conditions.

For SAM's, kinematic performance is dependent on the speed, altitude and manoeuvrability of the target. For AAM's, there exists the further complication, for modelling purposes, of there being a dependence on the speed and altitude of the launch aircraft.

The very large number of characteristics to be modelled in this environment and these types of objects so as to provide adequate fidelity, has in the past meant that there was a need for a large number of sets of graphs which could be used manually or by computer, to predict the SAM and AAM flight path for a selected number of initial conditions. When initial conditions lay between available graphs, users extrapolated the likely results.

Not only were the number of initial conditions many and thus the number of graphs unwieldy to handle, but, when the threat can apply to a large variety of aircraft such as jets (F/A-18, F111-C), slower aircraft like transports (C-130, Caribou) and helicopters, the prior system was very difficult to use and equally difficult for defence personnel to maintain. These difficulties were exacerbated when missile characteristics changed with technological improvements as that necessitated the issue of updated graphs for all the anticipated contingencies.

As computers became more widely used in the military the unwieldy nature of the many graphs began to disappear. A computer allows a very large amount of information to be stored and made almost instantaneously accessible.

Having a computer though, requires specialised programs to provide missile simulation at different levels of fidelity.

The ways in which known computer based simulators are implemented involves the use of either a general purpose or dedicated computing device, sometimes dedicated hardware and software supplied with initial and scenario related data, sometimes including classified missile characteristics. All of this data is called upon by the program(s) running on the computer device as required. Key words in the data file are used to call associated data.

Simulation programs of this type provide a dedicated calculation function as well as dedicated display functionality so that the simulation can be illustrated. Simulations which use precompiled software routines are preferable as this improves the speed of the large number of calculations involved in simulating high speed and complex devices such as aircraft and missiles.

If a simulator provides a simulation of one particular SAM then its characteristics would be predetermined and could be precompiled into the code of the program. If enough of the various calculations are executable from the precompiled code

simulation speeds can be optimised to suit the most demanding of scenarios as well as being able to handle various real time inputs and outputs provided by the simulation participants.

Some prior simulators recognise this advantage and are designed to provide a mechanism for reusing the common compiled elements. Unfortunately such an approach also requires the use of large amounts of uncompiled code and data, which may adversely affect the computation efficiency, and in general raises the need for raw computing power.

In addition to the compiled code to the computing power trade off, there is also a fidelity issue. The very complex calculations that take place are algorithmic and iterative and the higher the order of the iteration, the higher the accuracy of the results. If however, the computing power to perform high numbers of iteration is limited in any way, accuracy can be affected and the simulation less useful.

Furthermore, computing power availability can vary during a simulation, thus the fidelity of the results may also be affected as the number of iterations may be dynamically limited to allow a particular computing device to handle all the simulation tasks.

As an example of the unpredictability of the computing load and the potential complexity of the simulation task, a good simulation should not only accurately predict a flight path, it should be able to replicate real time anti-missile measures such as Electronic Warfare (EW) jamming and its effect on the missile.

Not only do these effects need to be applied on-the-fly, they also need to be able to respond to the anti-EW measures available to the missile and also any user input occurring at the same time.

Such a demand on the simulator is not predictable in time or duration and to adequately account for all possible situations requires that there be vast computer resources available at all times.

One way of handling such limitations is to reduce the zone within which the simulation can occur and/or within which the parties to the simulation are able to detect certain missile events.

This compromise however, limits the reality of the simulation and trade offs of this type need to be carefully applied, as exceptions in a simulation may adversely affect the ability of the trained personnel to handle real life situations.

Thus the challenge for the designers of this invention was to create an architecture for simulation which took advantage of the available technology but which avoided or minimised the problems of prior simulation models, particularly the need to have vast quantities of processing speed, power and resources while creating an easily maintained and secure database of missile and aircraft characteristics for military use, although in developing the invention it has become apparent that the modelling technique described herein has many possible applications, and is not restricted to missile engagement and coverage analysis.

Fig. 1 depicts a schematic of the various models which combine to comprise an embodiment of the invention.

A Graphical User Interface (GUI) receives and sends data from and to the Engagement Model. The GUI can be configured to work on a variety of computer operating systems and in a preferred embodiment the GUI is designed to execute in a Microsoft Windows™ environment (eg Windows 95/NT).

The functional elements of the GUI are preferably provided by menu options and virtual buttons as well as context relevant help.

A further preferable feature for a user is the provision of predetermined context relevant default values in the various fields to be set by the user.

In addition to which it is possible for the user to use a variety of quantitative measurement units such as kilometres, metres and nautical miles for distance.

It is also a preferred feature that the results of the missile engagement and coverage analysis (MECA) software and method can be imported into the Microsoft Windows™ applications in the form of graphs and/or data which can be imported to spreadsheets. Any of the graphical output of MECA can be cut and pasted into the Windows Clipboard. For example, a particular plot can be cut and pasted into a Microsoft Word document for reporting or off-line analysis.

It is also useful for the MECA GUI output to include two dimensional (2D) and three dimensional (3D) graphical views which can also be provided to various windows on one or more screens for operator guidance.

The GUI is not a required element since the Engagement Model which comprises a software model, is capable of running by itself and applying various equations to predetermined data representing the attacker, the target and the missile characteristics to produce the kinematics of each object. Missile data is loaded into the Engagement Model which also has a numerical integrator to integrate the various kinematics and a virtual data bus for facilitating the sharing of variables used in the simulation calculations.

In this embodiment the GUI is used to collect initial scenario data from the user so that the Engagement Model can use that data. The Engagement Model is, in this embodiment, run as a C++ class and is an object containing the required processing code such as in the various equations. Thus the Engagement Model is initiated by the GUI program at start-up and the class propagates the missile, attacker and target states using a predetermined integration scheme. The scenario is run in lock step

with the timing units determined by the simulation and applied throughout the scenario.

In this embodiment, the Rung-Kutta 4th order integration method is used but others are available and new ones can be added as required.

The Missile Model, in this embodiment has a 3 Degrees Of Freedom (3DOF) missile model having a predetermined structure with flexible parameter definitions. It has been designed generally so that different missiles can be simulated by merely applying different numeric parameters. 6DOF models are also capable of use in this embodiment but are not described herein. Defence Force operational requirements are generally satisfied by the 3DOF model.

The Missile Model can be run using numeric values received from by the data bus, built-in options or completely user-defined parameters.

The output of the Missile Model is provided to the Engagement Model and integrated into the scenario along with the attacker and target states.

The Engagement and Missile Models can operate stand-alone and provide useful output sufficient for some simulations because of their modularity and the fact that they are unclassified as they do not contain any sensitive data relating to the performance of missiles or aircraft.

For simple simulations and analysis, missile data files are supplied with the MECA program as unclassified data and tend to include publicly known missile parameters. However, when the missile data file contains sensitive missile parameters it will be classified and depending on its classification, the missile data file may be encrypted and require special physical handling. For example, a classified missile data file may not be used on untrusted computers or computers which are not within a closed and controlled computer network environment.

Clearly, defence forces are used to handling classified information and have suitable, additional environments for operating the program.

A missile data file can range in complexity from a simply structured text file to complex ALGOL-type programming language syntax. In this embodiment of the invention the data file comprises numeric data, syntactic key words and user defined algorithmic expressions (SMART DATA). The missile data files include a particular user-defined code block (algorithmic expressions) which, with the assistance of a processing action that, for example is provided by a parser and a virtual machine (sometimes referred to as a StacMac virtual machine), are processed on-the-fly so as to produce a compiled code block which is dedicated to calculating an appropriate value for a missile parameter. Fig. 1 is merely a schematic overview of but one embodiment of the MECA simulation program. The GUI permits various scenarios to be selected such as single engagement, range/velocity/latax (RVL), coverage boundary and performance contours to be selected.

In each of these scenarios the GUI facilitates the entry of various initialising parameters for the respective scenario.

In one example, in a Single Engagement scenario the GUI provides;

a missile selection box which can reference one of the available missile data files as previously disclosed;

attacker details such as its initial location (x, y, altitude);

velocity and heading; and

launch delay time in any concurrent manoeuvre (eg constant G's) by the launch aircraft.

Furthermore target details such as those of the attacker can similarly be entered by the user. Each entry required is provided default values for the increased useability of the program. For example in a Single Engagement scenario the GUI provides;

target details such as its initial location (x, y, altitude);

velocity and heading, azimuth, elevation, manoeuvre, horizontal latax, vertical latax and start manoeuvre delay.

The Engagement Model class includes member functions to set all of the appropriate parameters prior to stepping through the engagement simulation. Examples of the member functions are Specify the Missile Data File which contains all the missile specific parameters; Set Initial Conditions such as the Attacker/Target initial positions, velocity, altitude, manoeuvres and missile launch delay.

Simulation Controls such as integration time-step period, step simulation and run simulation to scenario completion are determined by a variety of triggers such as a preset time lapse or target destruction. Simulation Results follow the simulation run/completion or in the event of an error, these results can also indicate a hit or a miss of the target. Table 1 depicts some example of lines of code for initiating the Engagement Model.

Referring to Fig. 2 the output of the Engagement Model is supplied to the GUI and various displays of the engagement are available such as strip plots, 3D views from user definable view locations or even raw data for off-line analysis. In real time simulations, the engagement model provides data which is translated immediately into visual representations and feedback from the user of the simulation is in turn supplied to the engagement model attacker state and then sent to the relevant kinematic and integrated functions of the engagement model.

Fig. 2 depicts a schematic of the program modules which provide the functionality of the engagement class. The data manager loads missile data from the separate specified data file parsing it into an internal representation. The data is in effect, transferred from the file into the data manager but is made available to the rest of the software of the product via a virtual data bus as will be described later in the specification.

Thus, for example, the missile file syntax predetermines initialisation data such as mass, thrust, aerodynamic coefficients, latax capability and specific guidance laws. The data manager has read and write permissions for variables on the data bus and on an initial read provides those variables to the first of many kinematic equations which are central to the missile model and which is arranged to work with the Engagement Model.

The kinematic equations are applied and are also acted upon by the guidance laws applicable to the missile. Typically a predetermined number of guidance laws exist which are applicable to most missiles and each guidance law is precompiled and ready to operate given the required parameters.

This embodiment of the invention provides, as described previously, a means for user defined code blocks to be immediately parsed and used by a StacMac.

The parser/compiler within the data manager may compile code modules for use in the StacMac virtual machine as instructions which are executed and provide data back to the missile model. The calculated parameter values continue to be applied to the required kinematic equation within the time-step until the missile model stops or returns to the beginning of the process ready for the next time-step to commence. Having generated the applicable parameter values, they may be written by the data manager to the data bus so that other elements of the Engagement Model can use them as required.

Kinematic equations may provide as output the target, attacker and missile derivatives as well as the relative geometry between these three objects.

The Engagement Model may also include a control functionality which orchestrates the functions of the data manager and integration of each of the attacker, target missile states into the scenario in step with the time-step of the simulator and other time intervals relating to the virtual data bus and the GUI/interface.

As briefly described previously the data bus pictorially represented in Fig. 3 is actually a data structure rather than a physical bus which may be arranged to allow model variables to be shared amongst the many equations used in the missile and engagement model classes as well as between those classes and the data manager, user-defined code blocks and the GUI.

Fig. 4 depicts functional blocks of the missile model indicating that missile specific parameters are provided by the missile data file such as missile thrust, mass properties, aerodynamic coefficients and LATAX capabilities and applicable guidance laws. The Missile Model also accepts the results of the prior calculations and the relative kinematics of various objects.

Although the data flow arrows are depicted having particular orientations, such as guidance to air frame and propulsion to air frame and air frame to equations of motion, those shown are but one embodiment of an arrangement.

As depicted in Fig. 5 the data manager supplies data from the data bus and missile files to the missile model which contains the various equations which act upon the various parameters with the current time-step.

As described previously, one or more precompiled guidance laws may be applicable and when required user-defined guidance laws can be included in the missile files to account for a variety of situations. Such as for example when a known guidance law appears no longer to apply to a missile as a result of missile improvements, an estimate or a appropriate guidance law can be substituted in the missile file without having to change in any way the missile model or the engagement model of MECA.

Fig. 6 depicts an example of the operation of a parser which loads and verifies missile data files. The parser is arranged to identify in the missile data file (which is typically a text file) key words and numbers (eg NAVIGATION_GAIN 4.0). The parser also identifies user-defined code blocks and builds them into a representation that is executable by a StacMac.

Thus it will be appreciated that the missile data file can be a very flexible medium for defining missile parameters. Clearly, this flexibility is also going to be useful in modelling other environments and objects.

Not only is the file text based, the syntax used will be familiar to programmers and in terms of portability, transmission and security, it can be classified, encrypted and handled in ways familiar to military personnel.

In this embodiment, the user-defined code blocks contain ALGOL-type statements and have read access to all model variables and write access to the specific variable being calculated. At any one time the StacMac executes the user-defined code which assigns a value to the missile parameter.

Thus, using the code example provided in Fig. 6, the parameter, THRUST is required by the model. Values of THRUST could have been available from a look-up table versus time plot but, in this example, the guidance law parameter of THRUST is a user-defined block of code. The user-defined block starts and finishes with the key words Begin and End and contains ALGOL-type statements. The user-defined block has Read access to model variables such as vM (missile velocity) and must be able to Write to a model variable, in this example, THRUST SL (missile sea level thrust).

In a one-on-one simulation as pictorially represented in 2D in Fig. 7 two aircraft and two missiles are involved. Two objects of class TAircraft (Red and Blue) and two objects of class TMissile are handled in the simulation. Each T missile object is initiated with data from a respective missile data file to simulate its characteristics.

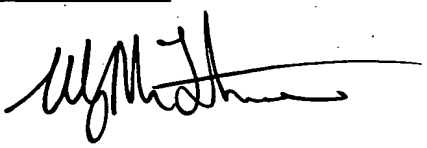
As shown in Table 2, the red missile is propagated through the class by passing the blue aircraft state as its target and the blue missile is propagated by passing the red aircraft state as its target.

As stated previously, the use of a MECA environment for the simulation is merely a convenient and useful example of the broader principle used in creating a simulation model that allows for the modelling element to be generic and for increased fidelity to be provided by creating an architecture which allows the use of "SMART DATA" comprising user defined modelling code unique to the object being modelled as it interacts with the scenario environment.

It will be appreciated by those skilled in the art, the invention is not restricted in its use to a particular application described and neither is the present invention restricted in its preferred embodiment with regard to the particular elements and/or features described or depicted herein. It will be appreciated that various modifications can be made without departing from the principles of the invention, therefore, the invention should be understood to include all such modifications within its scope.

DATED this 20th day of April, 2000.

THE COMMONWEALTH OF AUSTRALIA
By its Patent Attorneys
MADDERNS



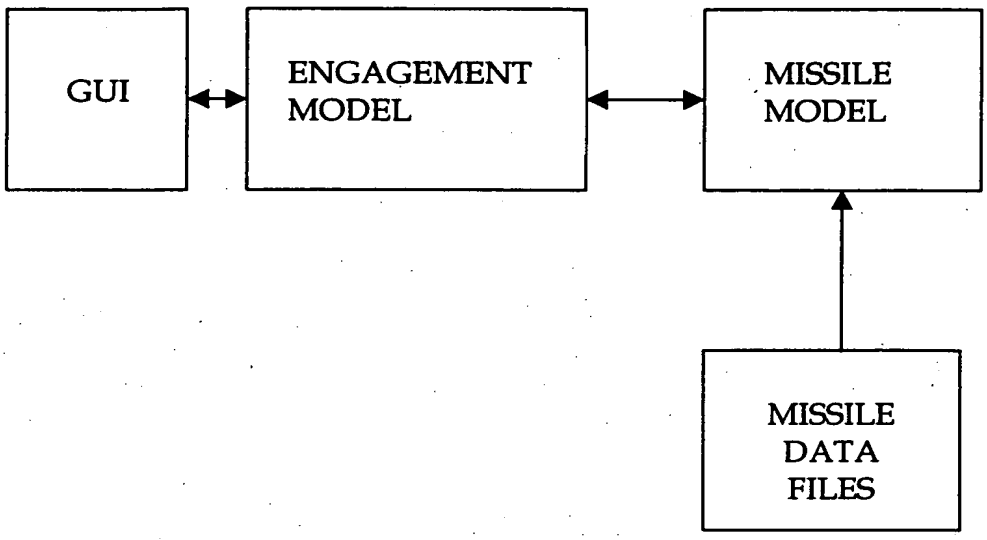

GUI
ENGAGEMENT MODEL
MISSILE MODEL
MISSILE DATA FILES

FIG. 1

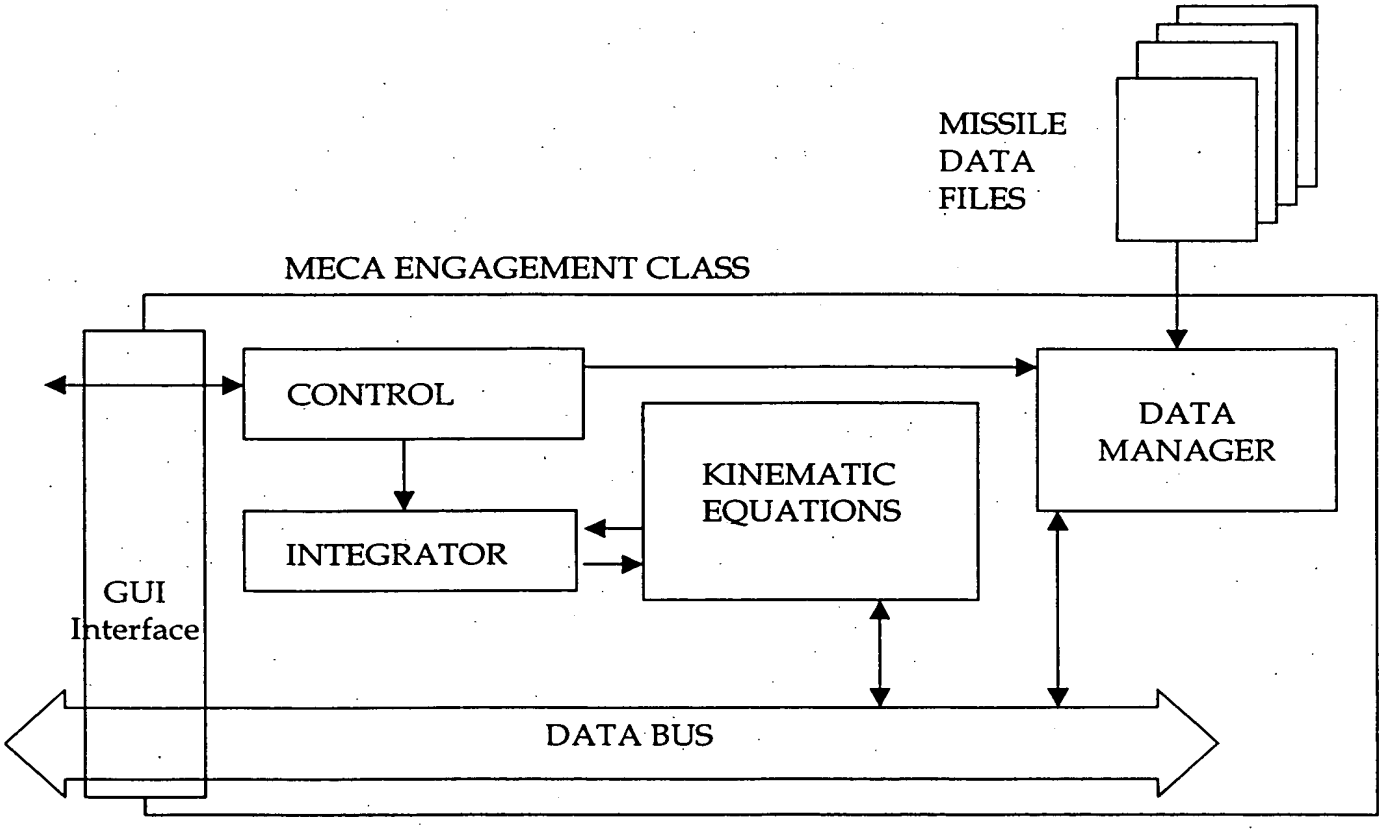

MISSILE DATA FILES
MECA ENGAGEMENT CLASS
CONTROL
DATA MANAGER
KINEMATIC EQUATIONS
INTEGRATOR
GUI Interface
DATA BUS

Fig. 2

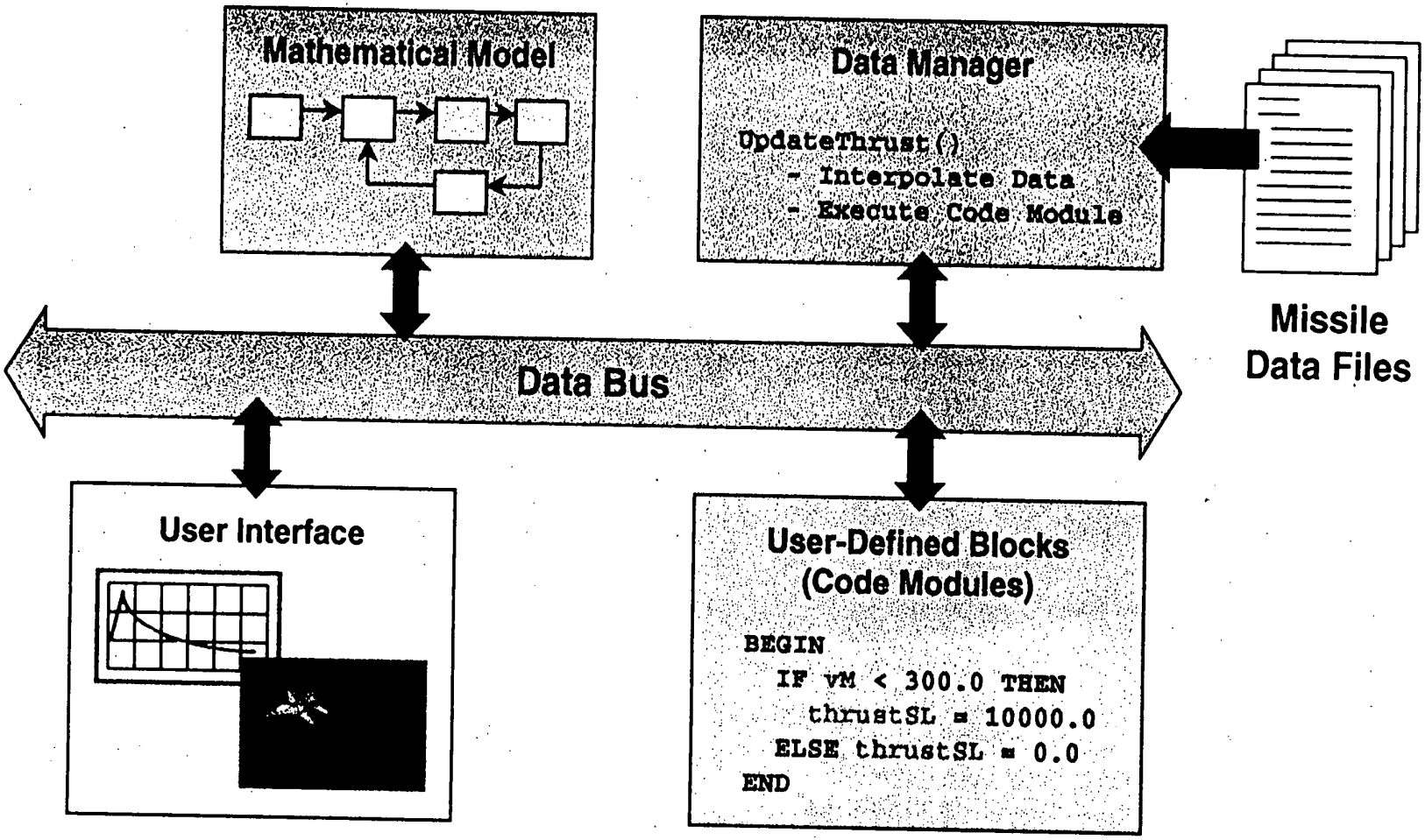


FIG. 3

BEST AVAILABLE COPY

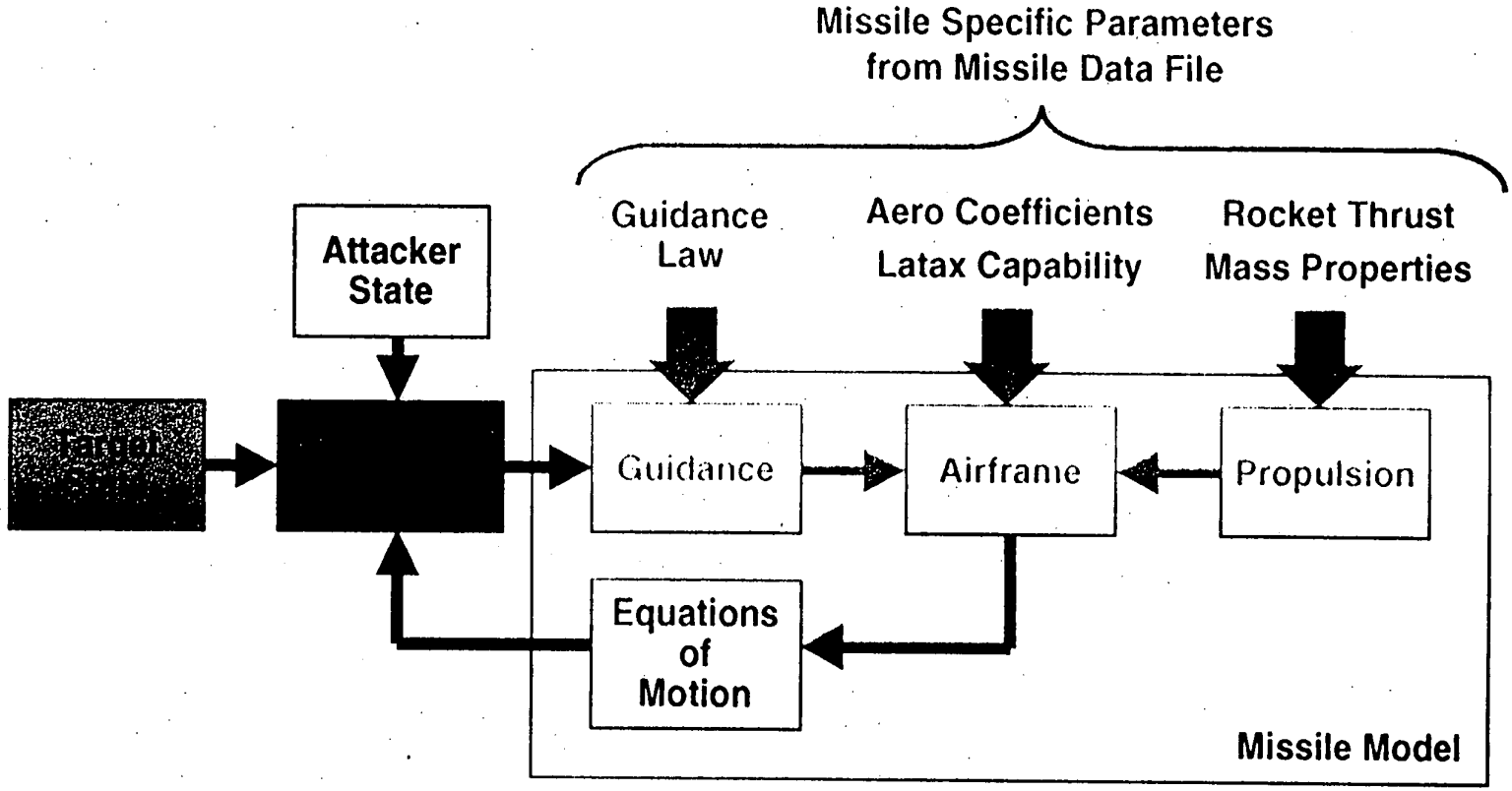


FIG. 4

BEST AVAILABLE COPY

Execution Process:

1. Data manager loads missile file, parsing it into an internal representation
2. Data manager updates initial state of data bus variables and internal model options
3. Model executes for each simulation time-step, calling user defined module where necessary via the StackMac virtual machine

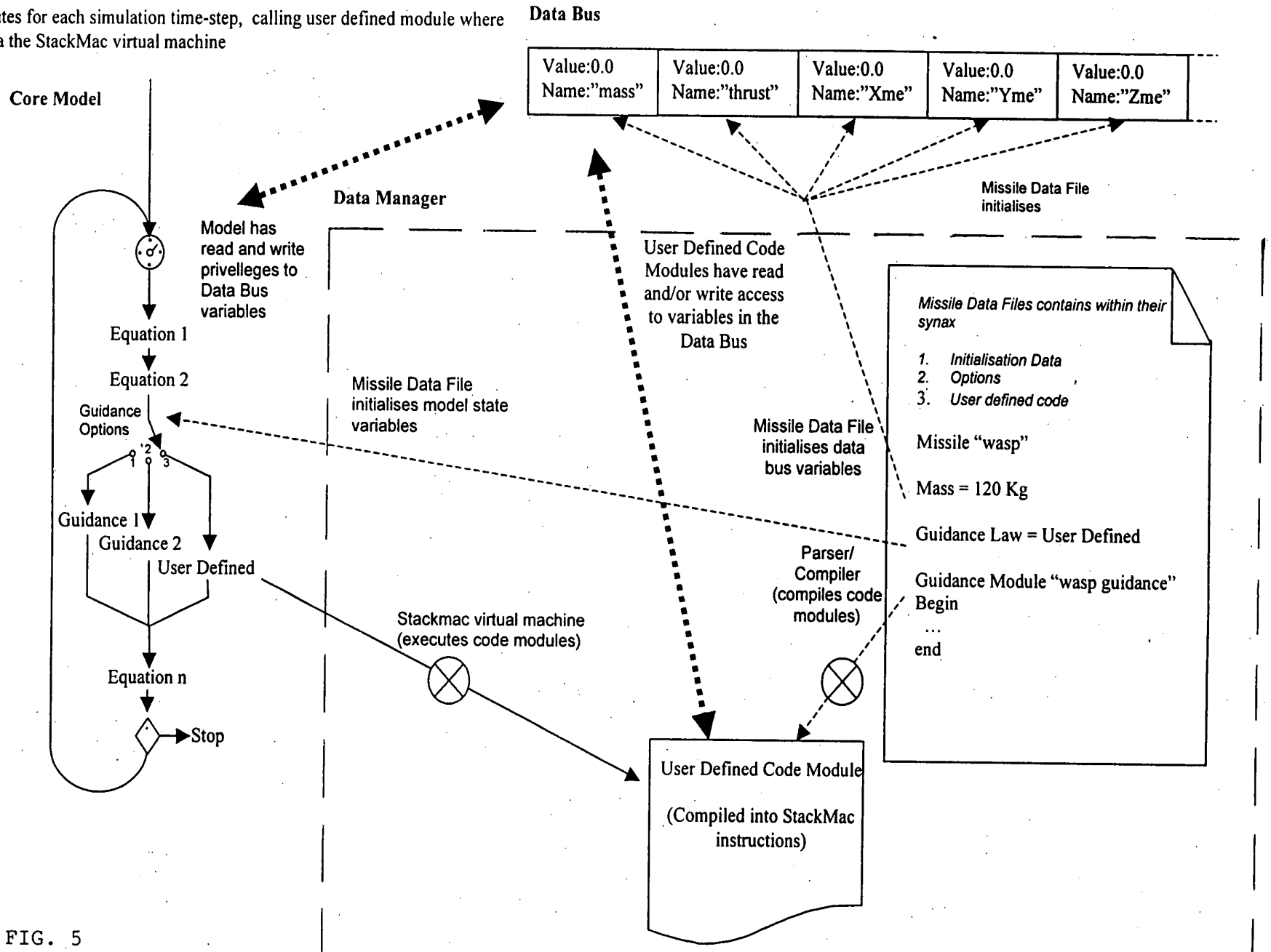


FIG. 5

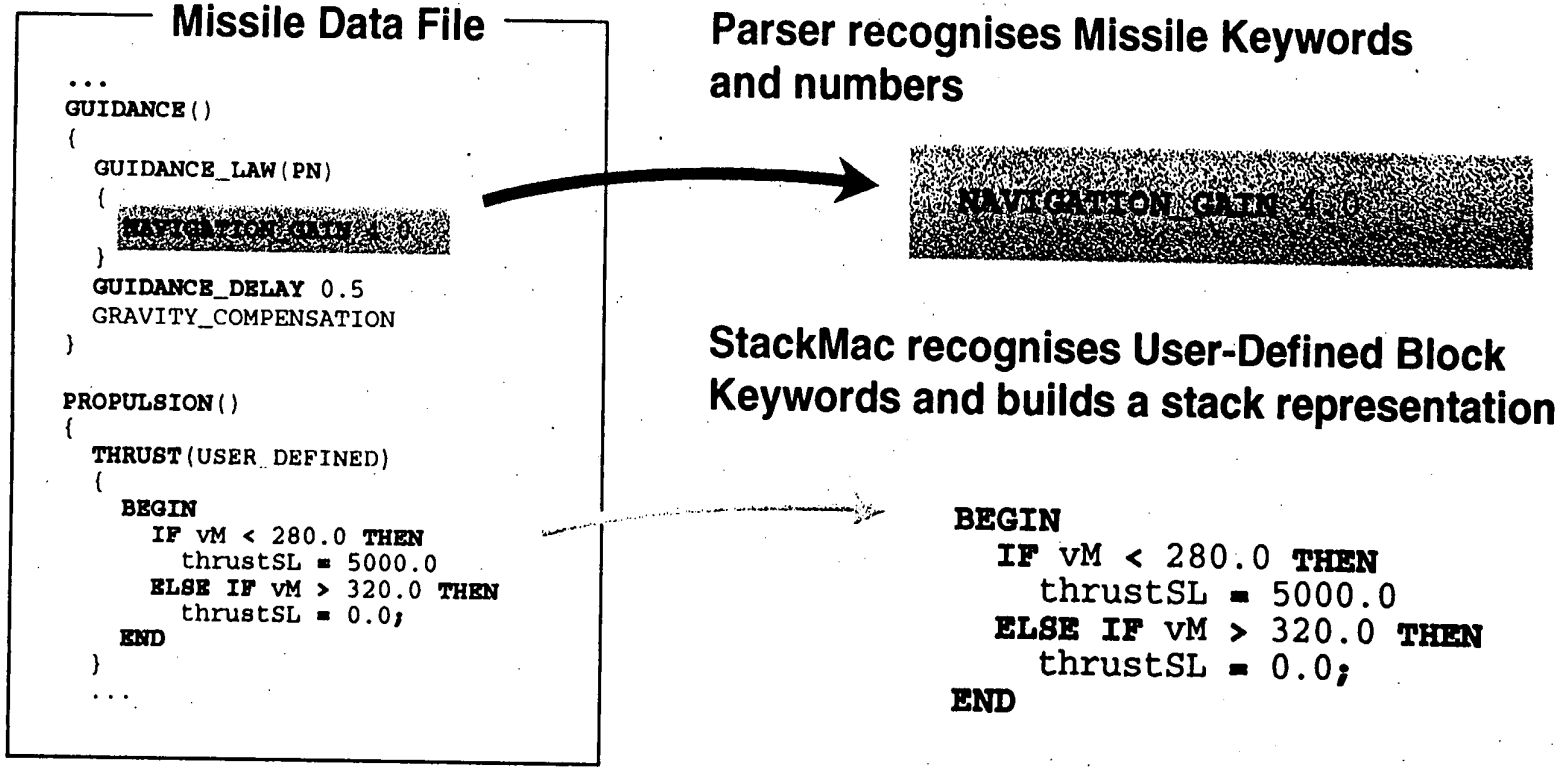


FIG. 6

BEST AVAILABLE COPY

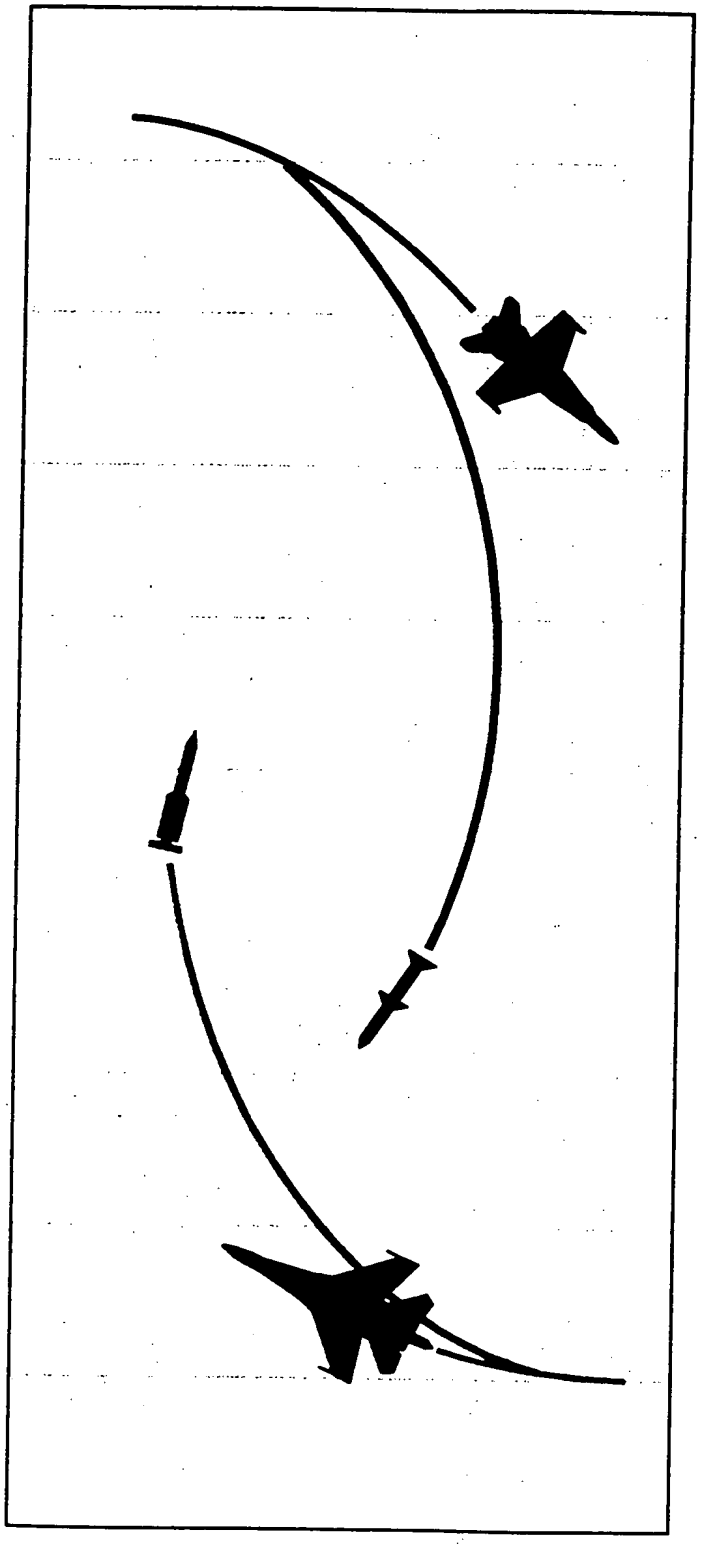


FIG. 7

BEST AVAILABLE COPY

Program startup

```
// Create Engagement Object
TEngagement  model;
...
```

```
// Set Attacker Parameters
model.SetAttackerPosition(x,y,z)
model.SetAttackerSpeed(v);
model.SetAttackerHeading(psi,theta);
...

// Set model time step
model.SetTimeStep(0.1);
...

// Run Engagement model
while (result != FINISHED)
{
  result = model.Step();
}
```

TABLE 1

Program startup

```
// Create Aircraft Objects
TAircraft  redAircraft;
TAircraft  blueAircraft;

// Create Missile Objects
TMissile  redMissile;
TMissile  blueMissile;
...

// Load Missile Data Files
redMissile.Load("AA-12.mis");
blueMissile.Load("AMRAAM.mis");
...
```

```
// Run Simulation
while (result != FINISHED)
{
  // Propagate aircraft
  redAircraft.Propagate();
  blueAircraft.Propagate();

  // Propagate missiles
  if (redLaunched)
   redMissile.Propagate(blueAircraft);

  if (blueLaunched)
   blueMissile.Propagate(redAircraft);

  // Check simulation STOP logic
  result = CheckStop();
}
...
```

TABLE 2